MY CI/CD PIPELINE CONTAINS ALL SECURITY TOOLS AVAILABLE! NOW WHAT...?

JUST ADDING MORE TOOLS WON‘T MAKE YOUR PRODUCTS MORE SECURE

Jasmin Mair
Elbsides November 2023

JASMIN MAIR

Global Product Security Manager @

Leica Microsystems

Connect with me:

jasmin.mair@owasp.org

linkedin.com/in/jasminmair

AGENDA

- Temptation to buy more tools
- What are teams really struggling with?
- Let’s figure a way out
- Implement tools responsibly
- Practical example

Credits: Ilya Pavlov on Unsplash

DANGER ZONE!

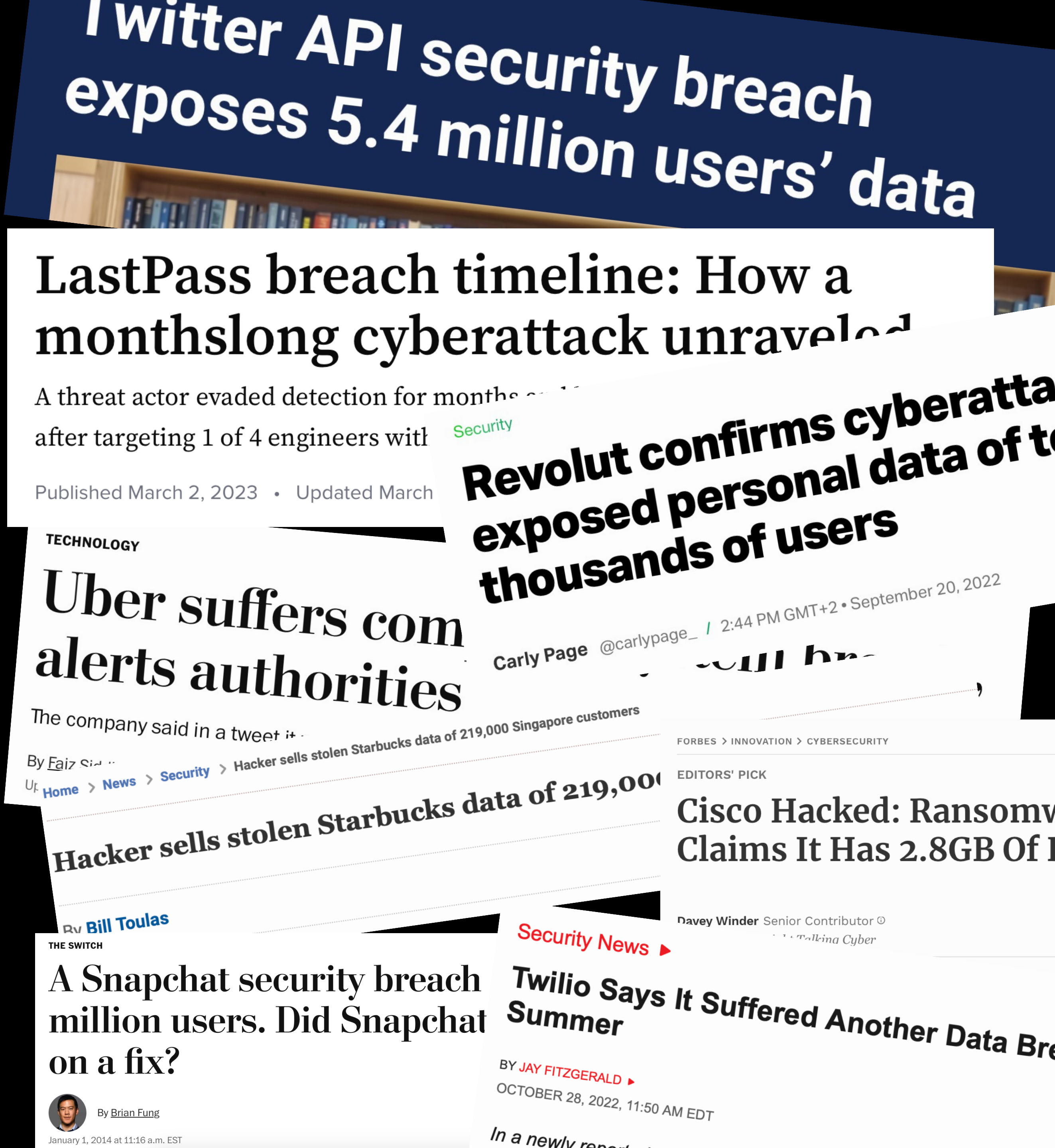
74% of all breaches include the human element[1]

26% of incidents were caused by exploiting public facing applications[2]

13% of attack vectors were vulnerabilities in third-party software[2]

[1] Verizon 2023 Data Breach Investigation Report

[2] 2023 IBM X-Force Threat Intelligence Report



Twitter API security breach exposes 5.4 million users' data

LastPass breach timeline: How a monthslong cyberattack unravel

A threat actor evaded detection for month

after targeting 1 of 4 engineers wit

Published March 2, 2023 • Updated March

Security

Revolut confirms cyberatta exposed personal data of t thousands of users

Carly Page @carlypage_ / 2:44 PM GMT+2 • September 20, 2022

TECHNOLOGY

Uber suffers com alerts authorities

The company said in a tweet

By Faiz Si

Home > News > Security > Hacker sells stolen Starbucks data of 219,000 Singapore customers

Hacker sells stolen Starbucks data of 219,00

By Bill Toulas

FORBES > INNOVATION > CYBERSECURITY

EDITORS' PICK

Cisco Hacked: Ransomw Claims It Has 2.8GB Of

Davey Winder Senior Contributor

Talking Cyber

THE SWITCH

A Snapchat security breach million users. Did Snapchat on a fix?

By Brian Fung

January 1, 2014 at 11:16 a.m. EST

Security News ▶

Twilio Says It Suffered Another Data Bre Summer

BY JAY FITZGERALD ▶

OCTOBER 28, 2022, 11:50 AM EDT

In a newly rep

LET‘S BUY SOME MORE TOOLS

IriusRisk

SECURE CODE WARRIOR

ThreatModeler

- Architecture review
- Technology and framework decisions

- Project roadmap
- Business value and requirements

- Configuration management
- Infrastructure provisioning
- Deployment strategy

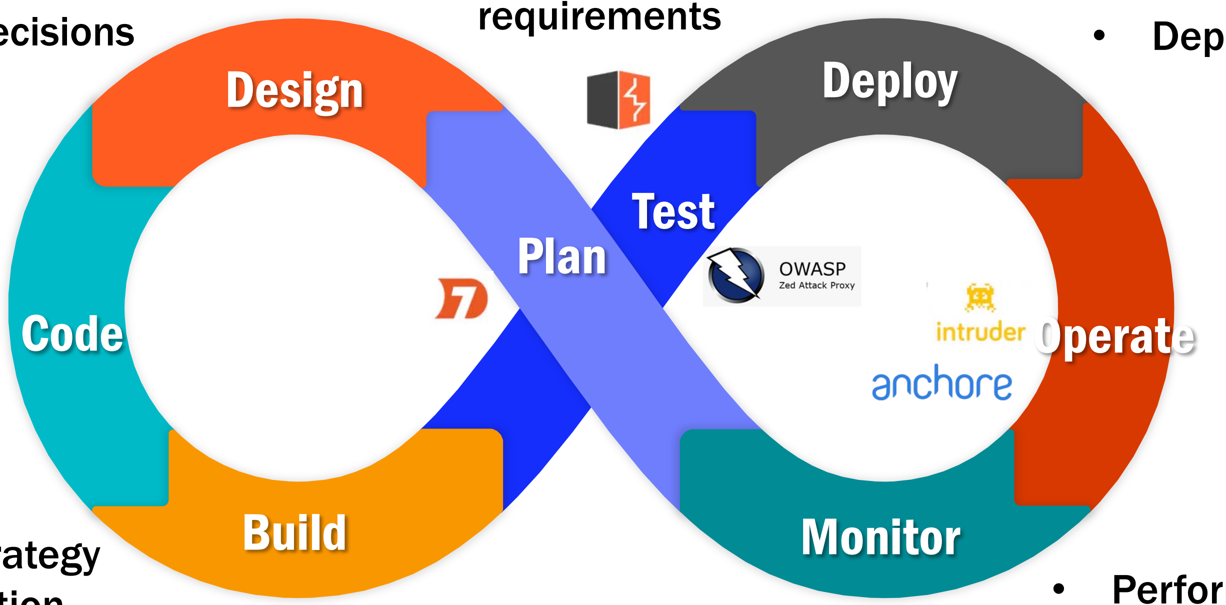
aqua

sysdig SECURE

StackRox

tenable

Falco



VERACODE

Checkmarx

- Code development
- Unit tests

- Scaling and configuration
- Bug and issue management
- Reporting on KPIs

Open Policy Agent

SYNOPSYS

WhiteSource

BLACKDUCK

DEPENDENCY-CHECK

snyk

HashiCorp Vault

KEYCLOAK

- Branching strategy
- Build application
- Artifact management

trivy

clair

- Integration tests
- E2E tests
- Performance tests

- Performance monitoring
- Infrastructure monitoring

DATADOG

Nagios

splunk>

GOTTA CATCH 'EM ALL

IriusRisk

SECURE CODE WARRIOR

ThreatModeler

sonarqube

- Architecture review
- Technology and framework decisions

Azure DevOps

JIRA

draw.io

- Project roadmap
- Business value and requirements

argo

Terraform

- Configuration management
- Infrastructure provisioning
- Deployment strategy

aqua

StackRox

sysdig SECURE

ANSIBLE

tenable

Falco

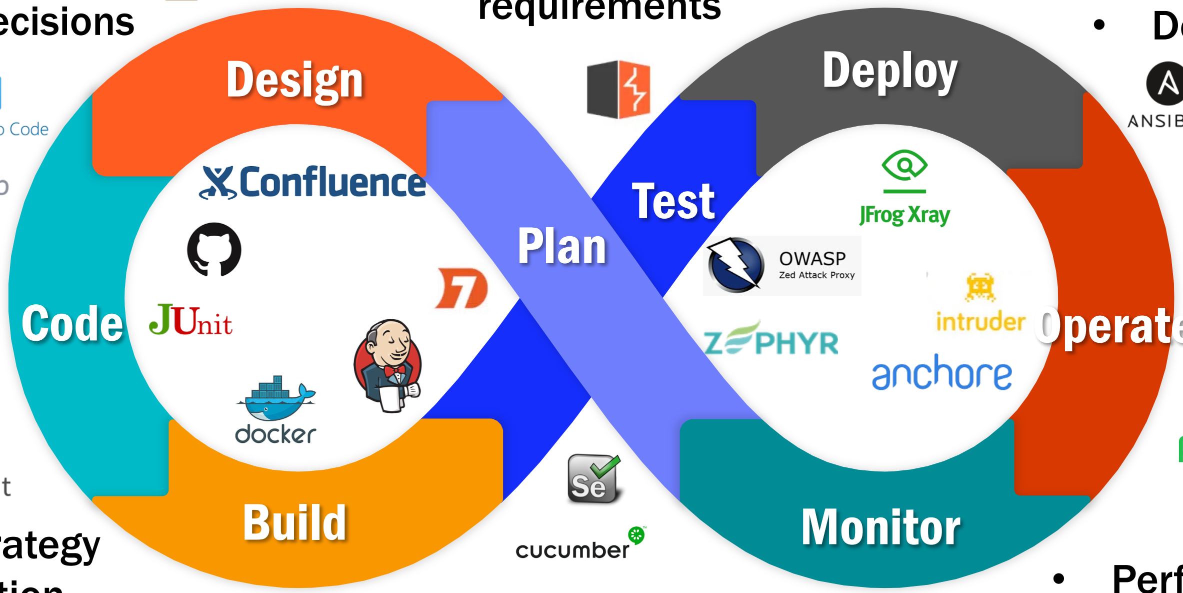
Visual Studio Code

GitLab

VERACODE

Checkmarx

- Code development
- Unit tests



- Scaling and configuration
- Bug and issue management
- Reporting on KPIs

Open Policy Agent

SYNOPSYS

WhiteSource

BLACKDUCK

DEPENDENCY-CHECK

ESLint

pagerduty

servicenow

RUNDECK

snyk

HashiCorp Vault

KEYCLOAK

- Branching strategy
- Build application
- Artifact management

Maven

clair

JFrog Artifactory

trivy

HELM

cucumber

- Integration tests
- E2E tests
- Performance tests

kibana

Prometheus

- Performance monitoring
- Infrastructure monitoring

DATADOG

Nagios

elastic

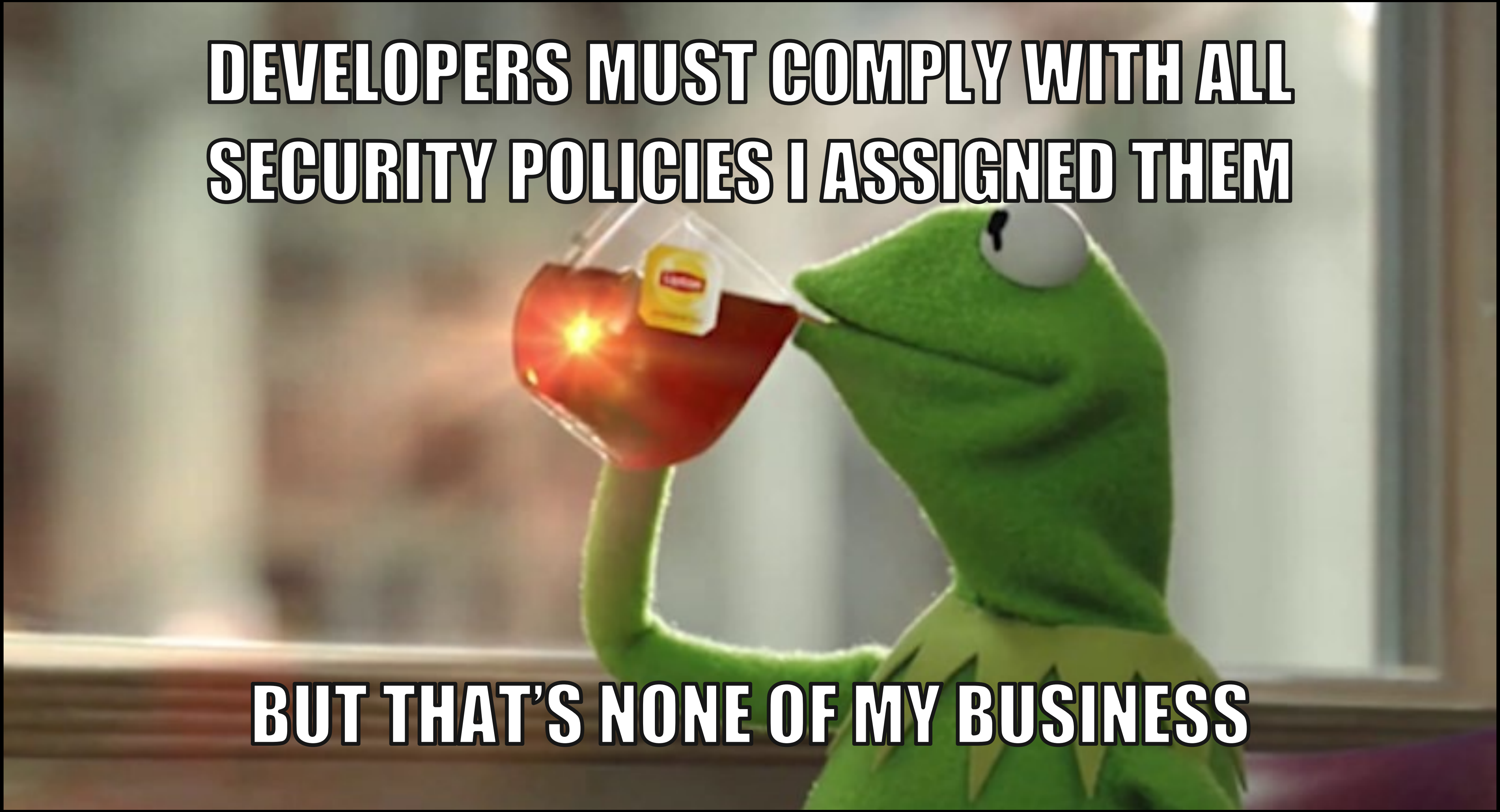
Grafana

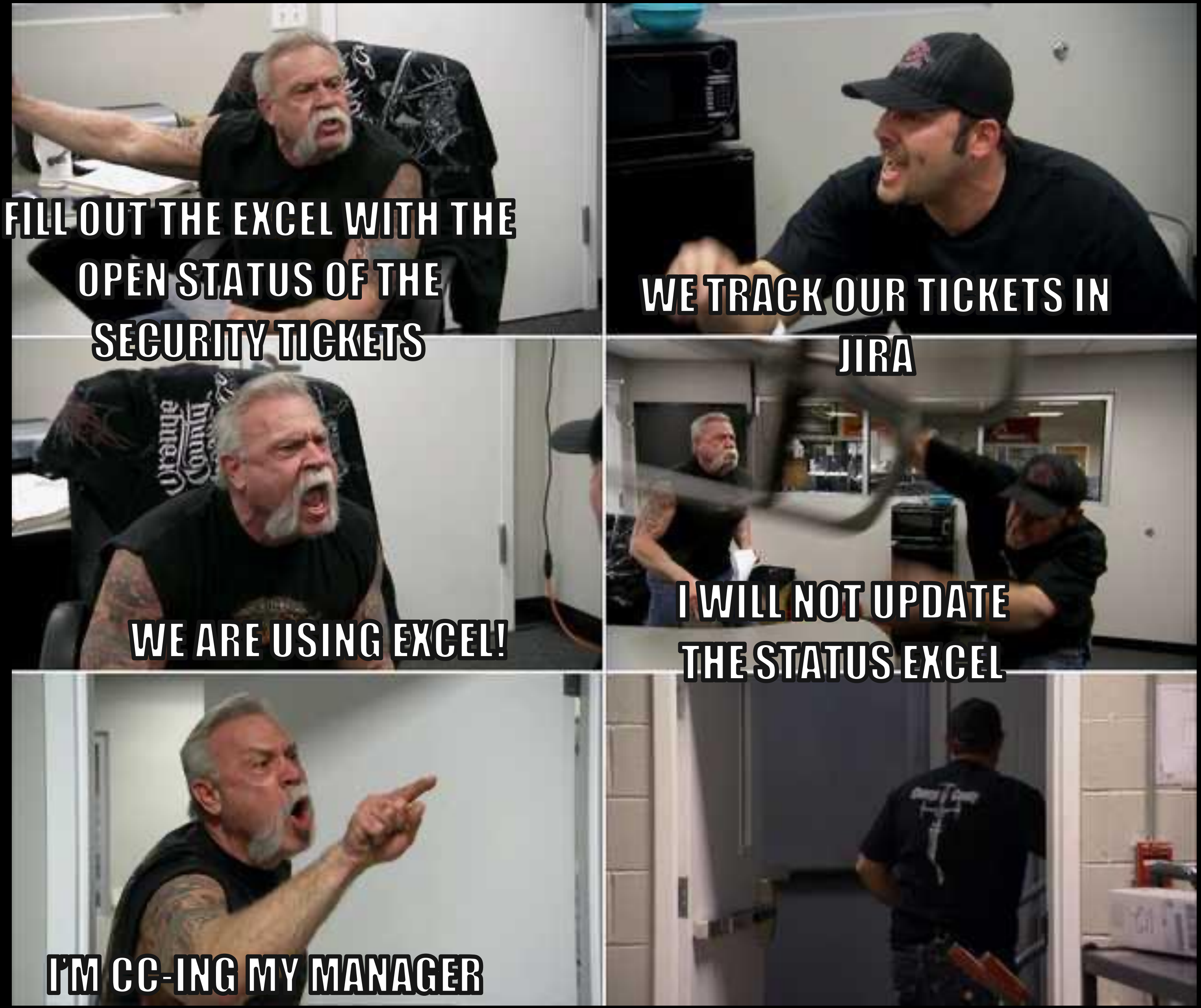
splunk>

ELBSIDES 2023

SECURITY
DEVELOPMENT

I SEE CYBER RISKS
EVERYWHERE


DEVELOPERS MUST COMPLY WITH ALL SECURITY POLICIES I ASSIGNED THEM
BUT THAT’S NONE OF MY BUSINESS


FILL OUT THE EXCEL WITH THE OPEN STATUS OF THE SECURITY TICKETS
WE TRACK OUR TICKETS IN JIRA
WE ARE USING EXCEL!
I WILL NOT UPDATE THE STATUS EXCEL
I'M CC-ING MY MANAGER

YOU CANNOT BUY DEVSECOPS

COLLABORATION CULTURE

- Align on security priorities
- Embed security experts in development team
- Clarify metrics and KPIs

AWARENESS & TRAINING

- Foster a security mindset among stakeholders
- Establish security champion program
- Train developers on security

SECURE THE SDLC

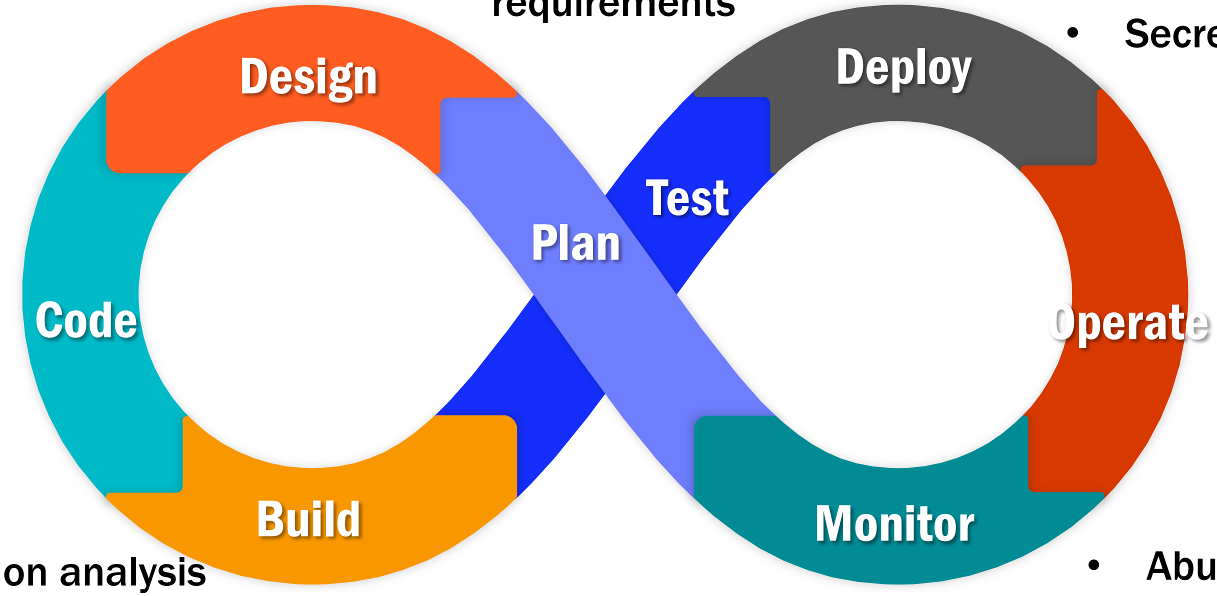
- Take security into consideration in each step of the SDLC
- Act on priorities and define security activities

WHERE TO GET STARTED

- Architecture and design review
- Threat modeling

- Application profiling
- Security requirements

- Application packaging
- Configuration management
- Secrets management



- Secure coding guidelines
- Security libraries
- Code reviews

- Runtime application security protection
- Vulnerability scanning

- Software composition analysis
- Container scans
- Image signature

- Static application security tests
- Dynamic application security tests
- Penetration tests

- Abuse case definition
- Security incident & event management

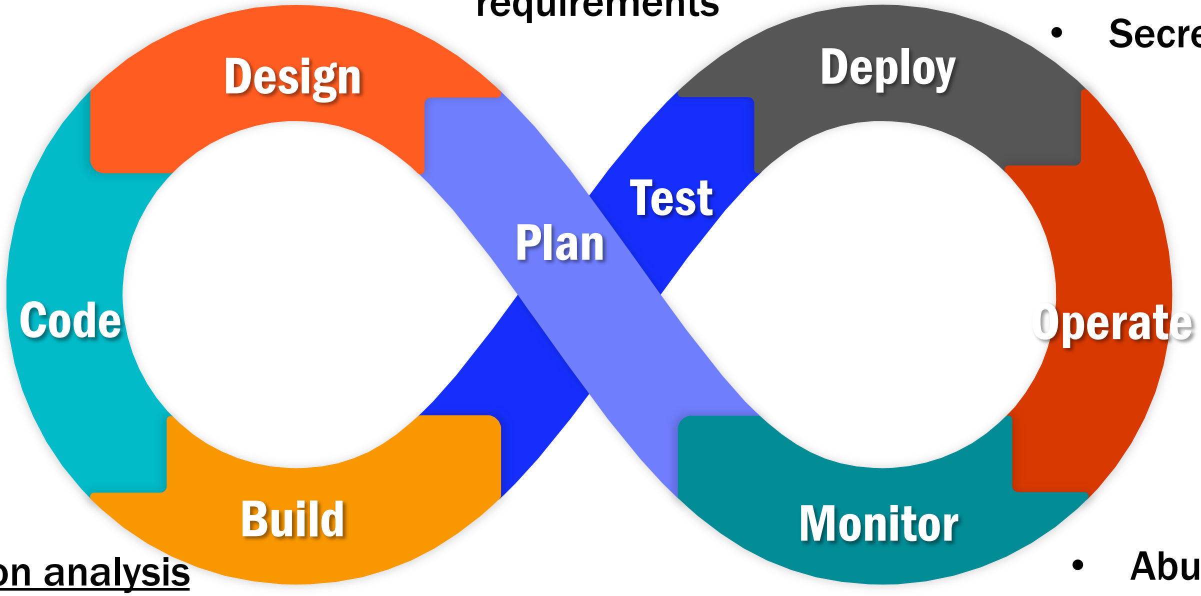
WHERE TO GET STARTED

- Architecture and design review
- Threat modeling

- Application profiling
- Security requirements

- Application packaging
- Configuration management
- Secrets management

- Secure coding guidelines
- Security libraries
- Code reviews



- Runtime application security protection
- Vulnerability scanning

- Software composition analysis
- Container scans
- Image signature

- Static application security tests
- Dynamic application security tests
- Penetration tests

- Abuse case definition
- Security incident & event management

ADDING ONE TOOL AT A TIME

TRAIN DEVELOPERS

- Clarify scope and goal of the tool
- Enable developers to manage findings
- Embed tool in CI/CD pipelines

SET BASELINE

- Scan existing code base
- Finetune scans
- Create meaningful quality gates

MANAGE FINDINGS

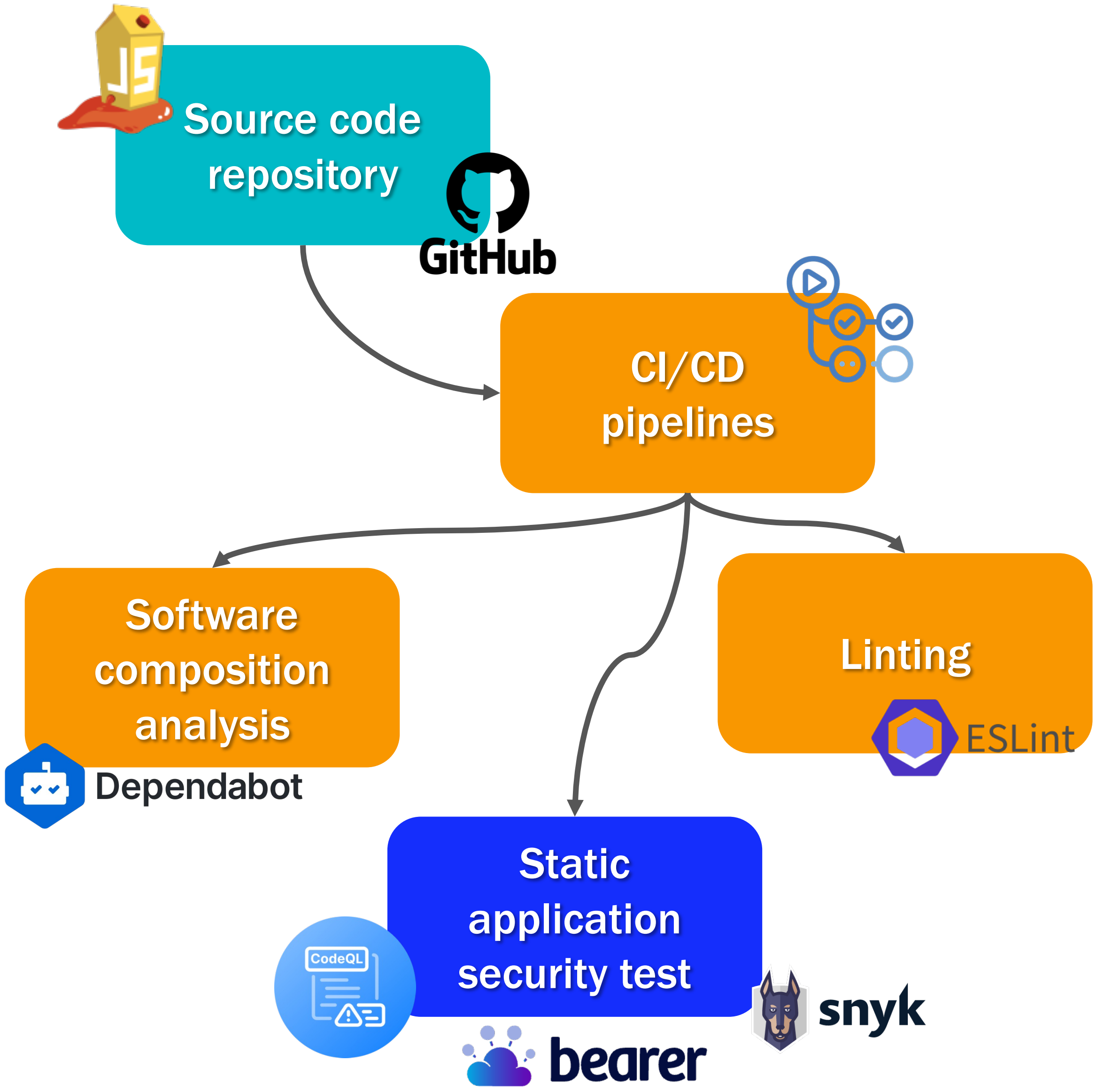
- Integrate tools with existing ticketing system
- Prioritize findings
- Visualize necessary work

INTRODUCING SAST

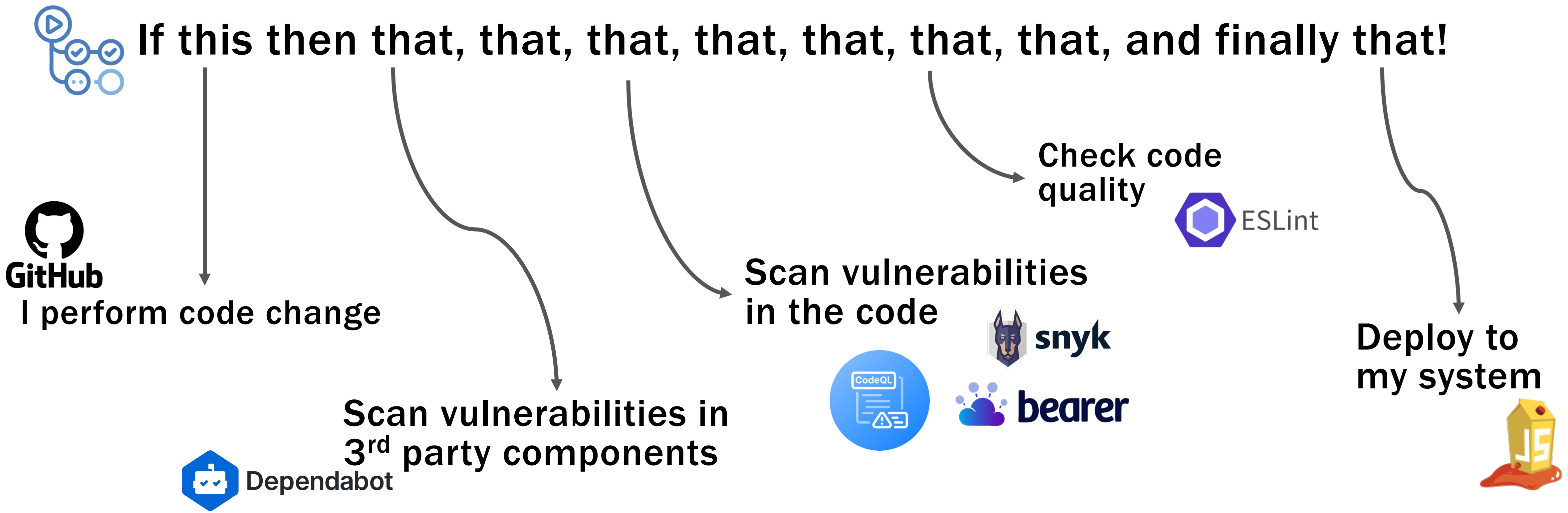
- Vulnerable web application: OWASP Juice Shop [1]
- Application hosted in GitHub [2]

[1] Copyright © by Bjoern Kimminich & the OWASP Juice Shop contributors 2014-2023

[2] https://github.com/jasminmair/sast-test-juice-shop



CI/CD SIMPLIFIED



If this then that, that, that, that, that, that, that, and finally that!

I perform code change

Scan vulnerabilities in 3rd party components

Scan vulnerabilities in the code

Check code quality

Deploy to my system

BASELINE SAST SCAN

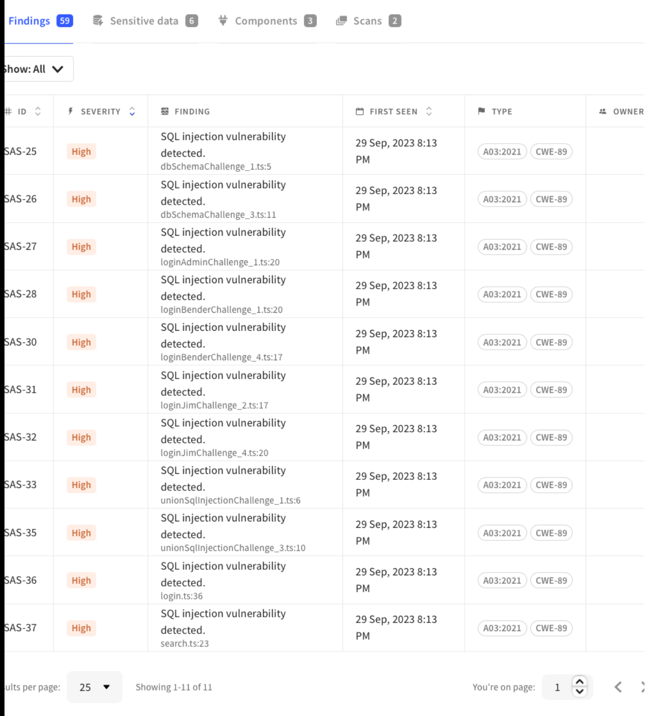
- Run a first manual scan
- Evaluate results
  - Get security status
  - Cluster findings by vulnerability type
  - Identify affected endpoints and areas of the application
- Exchange with a lead developer

Tool	Critical	High	Medium	Low	Total
Bearer	0	27	32	0	59
CodeQL	19	92	8	0	119
Snyk	0	30	32	219	281

CHALLENGES GALORE

- Discouraging amount of results
- Inconsistent results when comparing tools
- Complex output
- Results scattered over different places
- Integration in CI/CD pipelines

ELBSIDES 2023

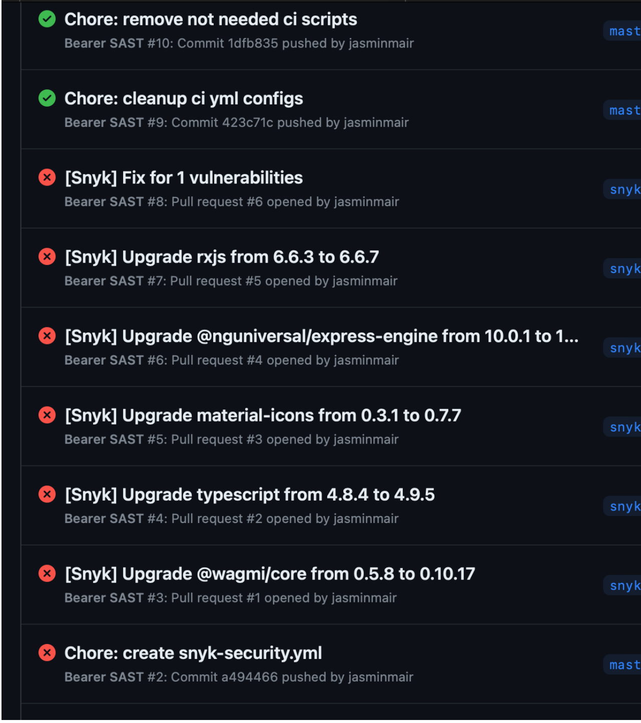


Findings 59 | Sensitive data 6 | Components 3 | Scans 2

Show: All

ID	SEVERITY	FINDING	FIRST SEEN	TYPE	OWNER
SAS-25	High	SQL injection vulnerability detected. dbSchemaChallenge_1.ts:5	29 Sep, 2023 8:13 PM	A03:2021 CWE-89	
SAS-26	High	SQL injection vulnerability detected. dbSchemaChallenge_3.ts:11	29 Sep, 2023 8:13 PM	A03:2021 CWE-89	
SAS-27	High	SQL injection vulnerability detected. loginAdminChallenge_1.ts:20	29 Sep, 2023 8:13 PM	A03:2021 CWE-89	
SAS-28	High	SQL injection vulnerability detected. loginBenderChallenge_1.ts:20	29 Sep, 2023 8:13 PM	A03:2021 CWE-89	
SAS-30	High	SQL injection vulnerability detected. loginBenderChallenge_4.ts:17	29 Sep, 2023 8:13 PM	A03:2021 CWE-89	
SAS-31	High	SQL injection vulnerability detected. loginJimChallenge_2.ts:17	29 Sep, 2023 8:13 PM	A03:2021 CWE-89	
SAS-32	High	SQL injection vulnerability detected. loginJimChallenge_4.ts:20	29 Sep, 2023 8:13 PM	A03:2021 CWE-89	
SAS-33	High	SQL injection vulnerability detected. unionSqlInjectionChallenge_1.ts:6	29 Sep, 2023 8:13 PM	A03:2021 CWE-89	
SAS-35	High	SQL injection vulnerability detected. unionSqlInjectionChallenge_3.ts:10	29 Sep, 2023 8:13 PM	A03:2021 CWE-89	
SAS-36	High	SQL injection vulnerability detected. login.ts:36	29 Sep, 2023 8:13 PM	A03:2021 CWE-89	
SAS-37	High	SQL injection vulnerability detected. search.ts:23	29 Sep, 2023 8:13 PM	A03:2021 CWE-89	

ults per page: 25 Showing 1-11 of 11 You're on page: 1



✓ Chore: remove not needed ci scripts
Bearer SAST #10: Commit 1dfb835 pushed by jasminmair — mast

✓ Chore: cleanup ci yml configs
Bearer SAST #9: Commit 423c71c pushed by jasminmair — mast

✗ [Snyk] Fix for 1 vulnerabilities
Bearer SAST #8: Pull request #6 opened by jasminmair — snyk

✗ [Snyk] Upgrade rxjs from 6.6.3 to 6.6.7
Bearer SAST #7: Pull request #5 opened by jasminmair — snyk

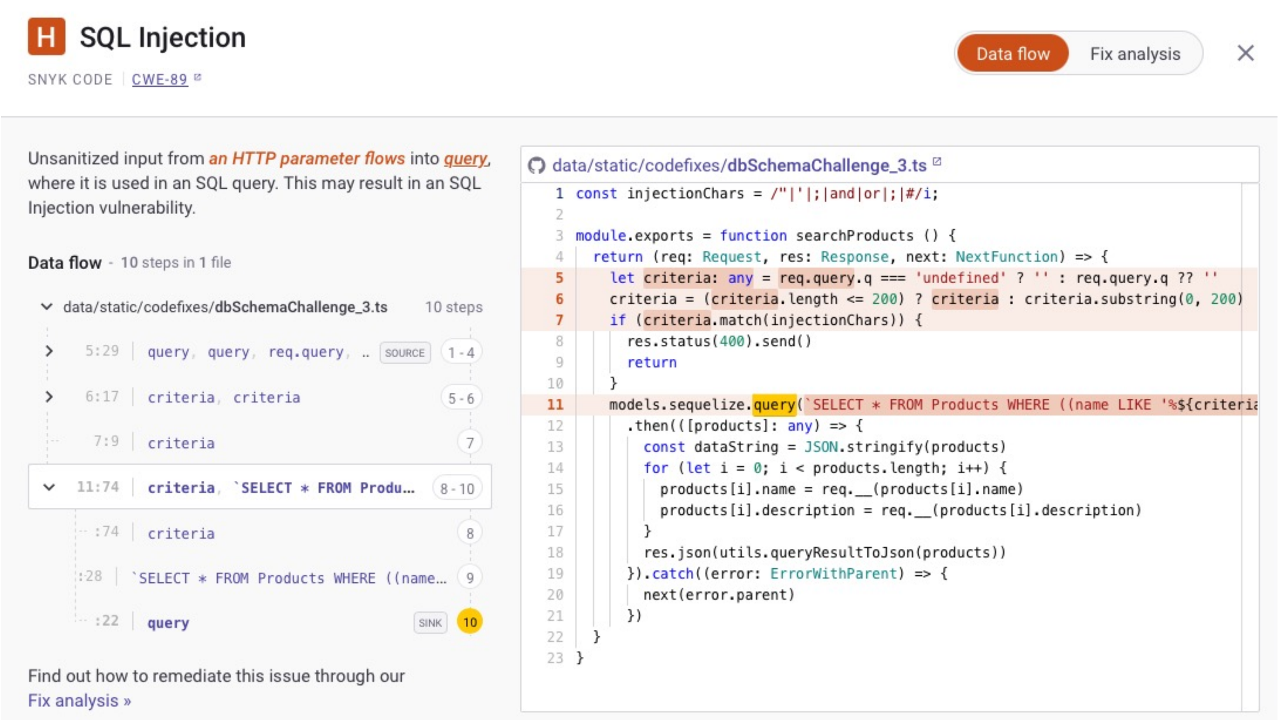
✗ [Snyk] Upgrade @nguniversal/express-engine from 10.0.1 to 1...
Bearer SAST #6: Pull request #4 opened by jasminmair — snyk

✗ [Snyk] Upgrade material-icons from 0.3.1 to 0.7.7
Bearer SAST #5: Pull request #3 opened by jasminmair — snyk

✗ [Snyk] Upgrade typescript from 4.8.4 to 4.9.5
Bearer SAST #4: Pull request #2 opened by jasminmair — snyk

✗ [Snyk] Upgrade @wagmi/core from 0.5.8 to 0.10.17
Bearer SAST #3: Pull request #1 opened by jasminmair — snyk

✗ Chore: create snyk-security.yml
Bearer SAST #2: Commit a494466 pushed by jasminmair — mast



H SQL Injection

SNYK CODE | CWE-89

Unsanitized input from *an HTTP parameter flows* into *query*, where it is used in an SQL query. This may result in an SQL Injection vulnerability.

Data flow - 10 steps in 1 file

data/static/codefixes/dbSchemaChallenge_3.ts — 10 steps

```
5:29   query, query, req.query, ..   SOURCE   1-4
6:17   criteria, criteria                     5-6
7:9    criteria                               7
11:74  criteria, `SELECT * FROM Produ...      8-10
   :74  criteria                              8
   :28  `SELECT * FROM Products WHERE ((name...   9
   :22  query                        SINK     10
```

Find out how to remediate this issue through our Fix analysis »

data/static/codefixes/dbSchemaChallenge_3.ts

```
1  const injectionChars = /"|'|;|and|or|;|#/i;
2
3  module.exports = function searchProducts () {
4    return (req: Request, res: Response, next: NextFunction) => {
5      let criteria: any = req.query.q === 'undefined' ? '' : req.query.q ?? ''
6      criteria = (criteria.length <= 200) ? criteria : criteria.substring(0, 200)
7      if (criteria.match(injectionChars)) {
8        res.status(400).send()
9        return
10     }
11     models.sequelize.query(`SELECT * FROM Products WHERE ((name LIKE '%${criteria
12       .then(([products]: any) => {
13         const dataString = JSON.stringify(products)
14         for (let i = 0; i < products.length; i++) {
15           products[i].name = req.__(products[i].name)
16           products[i].description = req.__(products[i].description)
17         }
18         res.json(utils.queryResultToJson(products))
19       }).catch((error: ErrorWithParent) => {
20         next(error.parent)
21       })
22   }
23 }
```

WHAT TO DO WITH FINDINGS?

SELECT A VULNERABILITY

- Select the highest priority vulnerability/flaw in the application to fix
- Decision is joint between business, development and security
- Raise awareness on the vulnerability

FIX VULNERABILITY IN CODE

- Train developers on how to mitigate the vulnerability
- Implement scanning tool on IDE level and in the CI/CD pipelines
- Fix the vulnerability in the code

CREATE QUALITY GATE

- Create a first quality gate in the pipelines to break the build if the vulnerability is identified again
- Select next vulnerability class and restart the process

SECURE PRODUCTS
DEVELOPMENT
SECURITY

Credits: Nora Carol Photography on Getty Images

THANK YOU

Jasmin Mair jasmin.mair@owasp.org